Eseguire il push di oggetti in blocco in FMC utilizzando REST-API

Sommario

Introduzione

In questo documento viene descritto come un amministratore API (Application Programming Interface) può eseguire il push di massa di oggetti Network, Port e URL in Firepower Management Center (FMC).

Prerequisiti

Requisiti

Cisco raccomanda la conoscenza dei seguenti argomenti:

- Informazioni su varie chiamate all'API REST. (Che cosa sono le API REST?)
- Revisione della Guida introduttiva all'API di FMC
- Revisione degli oggetti riutilizzabili di FMC
- Conoscenze base di Python richiede libreria

Componenti usati

- Firepower Management Center che supporta API REST (versione 6.1 o successiva) con API REST abilitata
- Interazioni API REST tramite Python.

Limitazioni

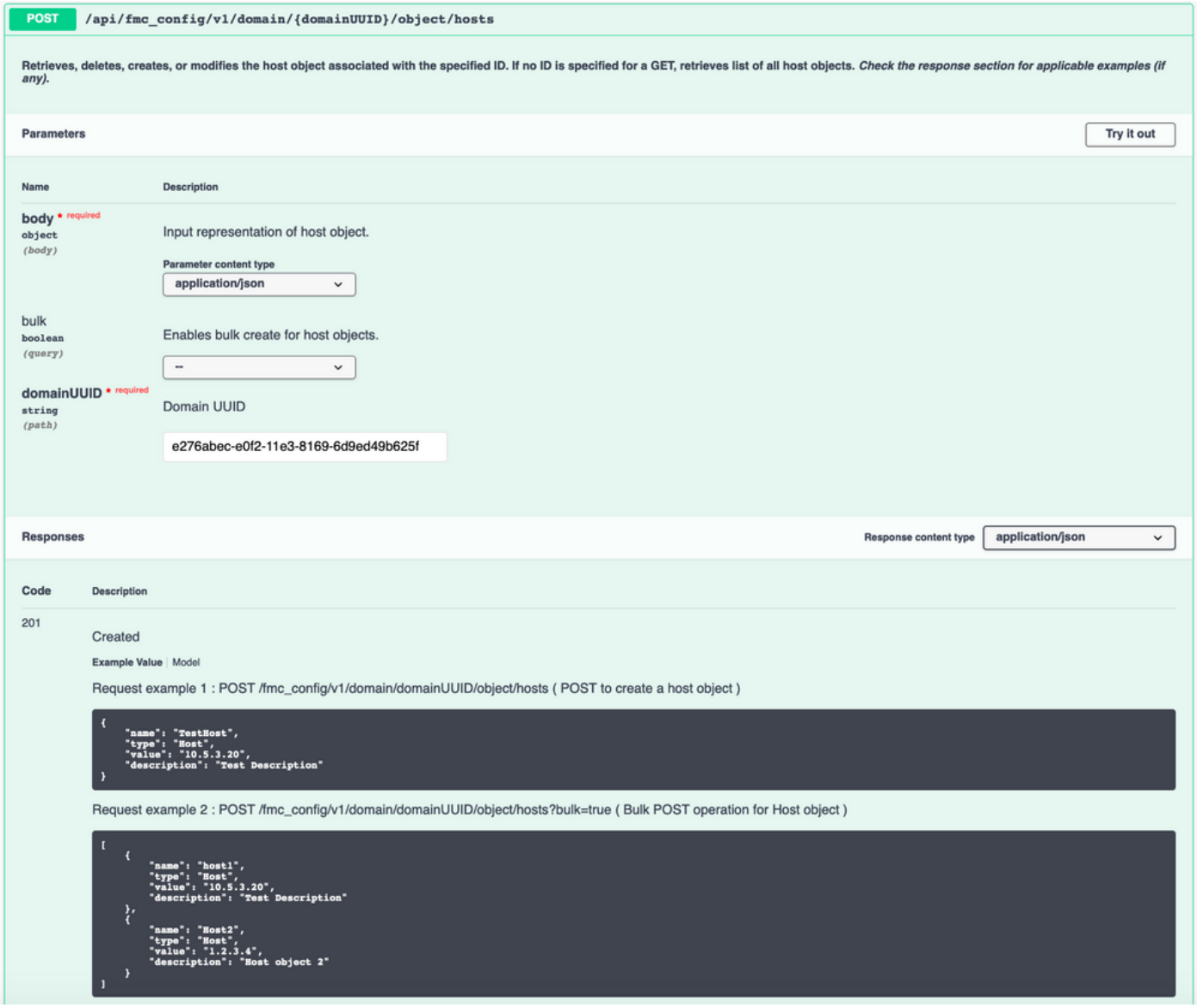
- Il nome dell'oggetto non può superare i 64 caratteri.
- Il nome dell'oggetto non deve contenere spazi all'inizio del nome e punti e virgola alla fine.
- Il payload non può contenere più di 1.000 voci in un singolo Bulk Push.
- Le dimensioni del payload non possono essere maggiori di 2 MB in un singolo Bulk Push.

Premesse

Le API REST sono sempre più diffuse a causa dell'approccio programmabile leggero che i gestori di rete possono utilizzare per configurare e gestire le loro reti. FMC supporta la configurazione e la gestione utilizzando qualsiasi client REST e utilizzando inoltre l'API explorer integrato.

Nell'esempio di questo documento viene utilizzato un file CSV come input e gli oggetti vengono inviati a FMC tramite l'interfaccia API REST. Il documento copre solo il Bulk push della rete host e una logica simile può essere estesa per gli altri oggetti. Al documento viene allegato un codice di esempio per gli oggetti URL e Port.

Di seguito è riportato il riferimento API per il POST sugli host di rete utilizzati, come mostrato nell'immagine:



Configurazione

Passaggio 1. Abilitare l'API REST e generare il token di autenticazione. Per istruzioni ed esempi di configurazione dettagliati, fare riferimento a Generate Authentication Token on FMC.

```
import requests import csv import json from requests.auth import HTTPBasicAuth from getpass import getpass address = input("Enter IP Address of the FMC: ") username = input ("Enter Username: ") password = getpass("Enter Password: ") api_uri = "/api/fmc_platform/v1/auth/generatetoken" url = "https://" + address + api_uri response = requests.request("POST", url, verify=False, auth=HTTPBasicAuth(username, password)) accesstoken = response.headers["X-auth-access-token"] refreshtoken = response.headers["X-auth-refresh-token"] DOMAIN_UUID = response.headers["DOMAIN_UUID"]
```

Passaggio 2. Convertire il file CSV fornito in un dizionario da utilizzare come payload JSON per la richiesta. Al documento viene allegato un file CSV di esempio per ogni tipo di oggetto.

	A	B	C	D	
1	name	description	type	value	
2	Host-1	Host-1	Host	10.10.10.10	
3	Host-2	Host-2	Host	10.10.10.1	
4	Network-1	Network-1	Network	10.10.9.0/24	
5	Host-3	Host-3	Host	10.10.10.2	
6	Range-1	Rannge-1	Range	10.20.20.1-10.20.20.20	
7					

```
csvFilePath = input("Please enter the CSV Filepath (For eg. : path/to/file/objects.csv) :") host = [] with open(csvFilePath, encoding='utf-8-sig') as csvf: csvReader = csv.DictReader(csvf) for rows in csvReader: if rows['type'] == "Host": host.append(rows) host_payload = json.dumps(host)
```

In questa fase, il valore di host_payload è identico a quello mostrato nell'immagine:

```
[{ "name": "Host-1", "description": "Host-1", "type": "Host", "value": "10.10.10.10" }, { "name": "Host-2", "description": "Host-2", "type": "Host", "value": "10.10.10.1" }, { "name": "Host-3", "description": "Host-3", "type": "Host", "value": "10.10.10.2" } ]
```

Passaggio 3. Creare la richiesta dall'input ricevuto dai passaggi precedenti e inviare la richiesta se il payload non è vuoto.

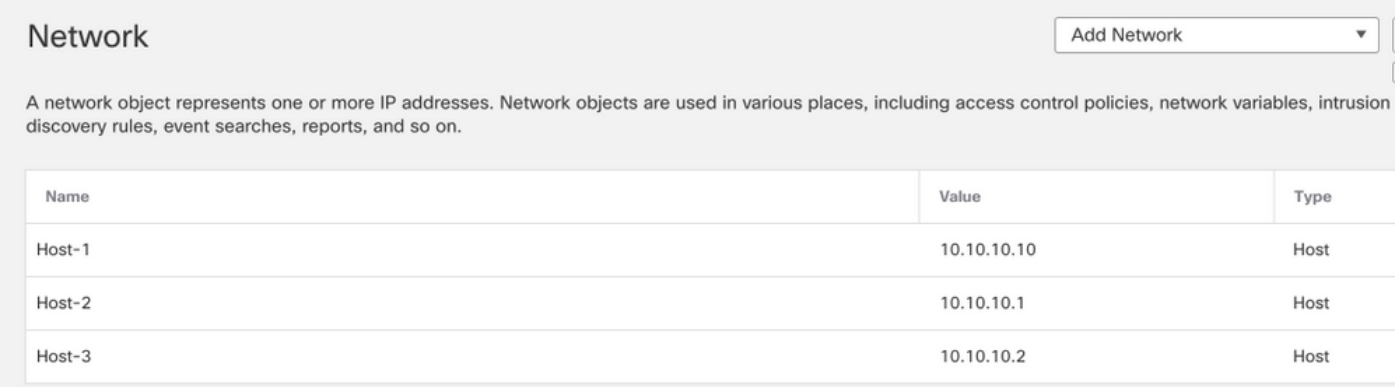
```
host_api_uri = "/api/fmc_config/v1/domain/" + DOMAIN_UUID + "/object/hosts?bulk =true" host_url = "https://" + address + host_api_uri headers = { 'Content-Type': 'application/json', 'x-auth-access-token': accesstoken } if host != []: response = requests.request("POST", host_url, headers=headers, data = host_payload, verify = False) else : print("Please Validate that the CSV file provided is correct or at correct location")
```

Verifica

- Stampare il codice di stato della risposta per verificare se la richiesta è riuscita o meno, come mostrato di seguito.

```
if response.status_code == 201 or response.status_code == 202: print("Host Objects successfully pushed") else: print("Host Object creation failed")
```

- Accedi a FMC Passare a **Oggetto > Gestione oggetti > Rete** e verificare gli oggetti host, come mostrato nell'immagine:



Risoluzione dei problemi

- Quando si utilizza il client REST, è possibile che vengano visualizzati errori correlati al problema del certificato SSL a causa di un certificato autofirmato. È possibile disattivare questa convalida a seconda del client in uso.
- I token di autenticazione dell'API REST di FMC sono validi per 30 minuti e possono essere aggiornati fino a tre volte.
- L'errore relativo alla richiesta può essere estratto dal corpo della risposta. Può essere raccolto come file di log per facilitare la risoluzione dei problemi.

```
logfile = "requestlog.txt" log = open(logfile,"w+") log.write(response.text) log.close
```

- Tutte le richieste REST vengono registrate in questi due file di registro in FMC. Cercare l'URL (ad esempio .../object/hosts) con l'operazione corretta. Se si sta cercando un errore per l'operazione GET, verificare che il log avvii un'operazione simile a GET...oggetto/host)

```
tail -f /var/opt/CSCOpx/MDC/tomcat/logs/stdout.logs tail -f
/var/opt/CSCOpx/MDC/log/operation/usmsharedsvcs.log
```